Is CoAP Congestion Safe?

Järvinen, Ilpo Johannes

ACM
2018-07-16

Järvinen , I J , Raitahila , I K , Cao , Z & Kojo , M P I 2018 , Is CoAP Congestion Safe? in ANRW '18 : Proceedings of the Applied Networking Research Workshop . Applied Networking Research Workshop , ACM , New York, NY , pp. 43-49 , ANRW '18: Applied Networking Research Workshop , Montreal , Quebec , Canada , 16/07/2018 . https://doi.org/10.1145/3232755.32328

http://hdl.handle.net/10138/325767
https://doi.org/10.1145/3232755.3232857

unspecified
acceptedVersion

Is CoAP Congestion Safe?

Ilpo Järvinen
University of Helsinki
ilpo.jarvinen@helsinki.fi

Iivo Raitahila
University of Helsinki
iivo.raitahila@helsinki.fi

Zhen Cao
Huawei
zhen.cao@huawei.com

Markku Kojo
University of Helsinki
markku.kojo@cs.helsinki.fi

ABSTRACT

A huge number of Internet of Things (IoT) devices are expected to be connected to the Internet in the near future. The Constrained Application Protocol (CoAP) has been increasingly deployed for wide-area IoT communication. It is crucial to understand how the specified CoAP congestion control algorithms perform. We seek an answer to this question by performing an extensive evaluation of the existing IETF CoAP Congestion Control proposals. We find that they fail to address congestion properly, particularly in the presence of a bufferbloated bottleneck buffer. We also fix the problem with a few simple modifications and demonstrate their effectiveness.

CCS CONCEPTS

• **Networks → Network performance evaluation**; *Cross-layer protocols*; Transport protocols; Application layer protocols;

1 INTRODUCTION

The Internet of Things (IoT) devices are expected to become ubiquitous in the near future and communicate over the Internet. Tens of billions IoT devices are expected to be connected to the Internet by year 2025. For any Internet traffic, congestion control is very important consideration as it is necessary to ensure stability and reasonable performance of the Internet. Safe handling of congestion is the primary objective of the congestion control. Any secondary performance related optimization in congestion control algorithms must be considered through its effects on reaching that primary object to ensure the stability of the Internet.

While the effect of a single device, especially in case of constrained IoT devices, may seem miniscule, it is inevitable that the expected, very large-scale deployment of IoT devices will incur high traffic volumes and congestion at least in some settings. Such traffic may impact the Internet or some parts of it in a harmful way unless the congestion control algorithms deployed on those devices are congestion safe, even in the presence of high density hotspots.

The Constrained Application Protocol (CoAP) [14] is the IETF standardized protocol for IoT. Many exchanges with CoAP are ephemeral which is challenging to any congestion control algorithm because it is difficult to acquire enough state to prevent persistent congestion that would cause severe problems. The CoAP base specification in RFC7252 includes only a very limited congestion control based on retransmission timeout (RTO) backoff. We call this congestion control Default CoAP. CoAP Simple Congestion Control/Advanced (CoCoA) [5] is an ongoing work that aims to improve CoAP congestion control algorithm such that persistent congestion is not caused by the CoAP traffic.

With the increasing deployment of CoAP and the trend of using it for Low-Power Wide-Area Network (LPWAN) [6, 13], it becomes important to understand how the CoAP Congestion Control proposals perform and if they are safe. This paper seeks an answer to this question by an extensive performance evaluation of the current specified algorithms. Our key findings and contributions include:

- We reveal that both Default CoAP and CoCoA fail to control congestion properly in a bufferbloated environment [7]. Particularly, they perform numerous unnecessary retransmissions that waste network capacity. This condition is stable and known as Congestion Collapse [11];
- We diagnose the current IETF proposals, and find that current CoAP congestion control specifications depart from the principles guided by Karn's algorithm [9] in significant ways, and as a consequence, pose a threat to the stability of the Internet;

ANRW '18, July 16, 2018, Montreal, QC, Canada

ACM ISBN 978-1-4503-5585-8/18/07…$15.00
https://doi.org/10.1145/3232755.3232857

- We propose an improved RTO backoff logic that allows Default CoAP and CoCoA to achieve robustness similar to TCP using Karn's algorithm. We confirm that the improved logic successfully solves the problem with unnecessary retransmissions, allowing the network to reach a safe operating point even under a very heavy load.

2 COAP CONGESTION CONTROL

The Constrained Application Protocol (CoAP) [14] has by default extremely basic congestion control based on an initial retransmission timeout (RTO) and an exponential backoff for the RTO timer. The initial RTO for each new reliable message exchange is set to a random value between two and three seconds to avoid synchronization effects and no RTT estimation is performed.

CoAP Simple Congestion Control/Advanced (CoCoA) [5] performs RTT measurements with a specific RTO calculation logic based on strong and weak RTO estimators. Both estimators are based on the TCP RTO calculation [12]. The strong RTO estimator is updated from RTT samples for the responses of not retransmitted requests. The weak RTO estimator is used otherwise but only if the request was not retransmitted more than two times. The weak RTO estimator uses 1 instead of 4 as the factor K for RTT variation. The current RTO is calculated from the RTO estimator that was updated most recently using exponentially weighted moving average with weight of 0.5 and 0.25 for strong and weak RTO estimator, respectively.

In addition, instead of using a binary exponential RTO backoff each time the retransmission timer expires, CoCoA uses a variable backoff factor; when the RTO is below 1 second CoCoA employs a backoff factor of 3 and when the RTO is above 3 seconds it employs the factor of 1.5. When the RTO is between 1 and 3 seconds, the backoff factor is 2.

CoCoA evaluations have shown that CoCoA performs better than Default CoAP [2–4, 8, 15]. However, heavily congested or bufferbloated settings are not covered by them.

3 TEST ARRANGEMENTS

3.1 Network

The test setup consists of IoT devices that communicate with a fixed host as shown in the Figure 1. The CoAP endpoints running on these hosts are implemented in libcoap [10]. The network connection is emulated with Netem (included in the Linux kernel) to have the characteristics of a wireless asymmetric link. The constrained bottleneck link between the last-hop router and the IoT devices has a data rate of 30 kbps downstream with 400 msecs one-way delay, 60 kbps upstream with 200 msecs delay and 296 bytes MTU. An additional 10 to 20 msecs delay with random variation is added between the last-hop router and the fixed host.

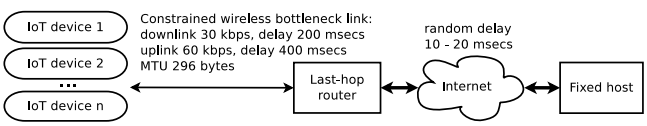


Figure 1: The test setup

The buffer size in front of the bottleneck link is selected from four options. The smallest buffer is 2500 bytes which is roughly the bandwidth-delay product of the link. The larger 14100, 28200 and 1410000 (infinite) bytes buffer sizes cause a varying degree of bufferbloat. The intermediate buffer sizes are omitted from the results when the buffer is large enough to no longer cause congestion losses.

3.2 Work Load

The number of concurrent IoT devices in each test varies from 1 to 400. There are two types of CoAP clients with only either one used in each test. The clients of both type represent typical CoAP traffic on IoT devices, exchanging small request-response pairs one at the time. The payload size in a CoAP response is 60 bytes. *Continuous* clients send requests until 50 successful pairs have been exchanged. *Random* clients divide the 50 pairs to random size batches of 1 to 10 pairs and after each batch all congestion control state information is reset. This emulates a scenario where multiple devices come and go one after the other, each exchanging only a few messages but 50 successful pairs are exchanged in total. The distinction is meaningful for CoCoA but not for ordinary Default CoAP, because it does not preserve any state information. Each testcase is run for 20 replications.

3.3 Congestion Control

We compare the congestion control algorithms Default CoAP as per RFC7252 [14] and CoCoA as per Internet-draft [5].

To prevent distorting the results, some of the CoAP parameters are modified. According to the CoAP specifications retransmitting is terminated after MAX_RETRANSMIT (by default four) retransmissions, but in our experiments with high congestion more retransmissions are required for a successful message exchange. Otherwise, a message exchange could become aborted, distorting the results as some clients would not be able to complete all 50 message exchanges. Thus, the MAX_RETRANSMIT parameter is set to 20 and the EXCHANGE_LIFETIME and MAX_TRANSMIT_WAIT parameters are modified correspondingly. To avoid unnecessarily long RTOs, the maximum backed off RTO is bounded to 60 seconds for Default CoAP.

Table 1: Flow completion times (secs) with 50 clients

CC	Buffer	min	10	25	median	75	90	max
Default	2500B	60.999	61.250	61.500	62.001	63.342	63.767	63.930
CoCoA	2500B	61.024	61.275	61.525	62.001	63.347	63.767	63.930
Default	14100B	62.077	62.177	62.377	62.690	63.003	63.178	63.304
CoCoA	14100B	62.077	62.177	62.377	62.690	63.003	63.179	63.304

4 RESULTS

The traffic over the bottleneck link is subject to congestion-related losses only. We include a router queue with different buffer sizes in front of the bottleneck link in order to experiment with the effect of a small buffer as well as larger buffer sizes representing different levels of buffer bloat that is typical in the various network devices deployed in the Internet today.

We evaluate the congestion control algorithms based on the following main metrics, namely, (i) flow completion time (FCT): the elapsed time to complete the exchange of 50 request-response pairs for a client, (ii) number of unnecessary retransmissions per client: the number of unnecessary retransmissions performed by a client while completing the exchange of 50 request-response pairs, and (iii) frequency of transmissions: the number of (re)transmissions needed for a successful exchange of a request-response pair.

4.1 Existing CoAP Congestion Control

With both 1 and 10 clients even the small buffer is large enough to hold all concurrent messages in flight. When only one client is transmitting, it completes the transmissions of 50 messages in 33.003 to 33.208 seconds. No congestion occurs and the flow completion time is limited only due to Round-Trip Time (RTT) of the network path that is around 660 msecs with one client and with 10 clients slightly longer due to a little amount of queueing. The queueing delay increases the flow completion time of 10 clients by up to a few hundreds milliseconds. There is no difference in results between Default CoAP and CoCoA, because no retransmissions were needed.

The flow completion times (FCTs) for 50 continuous clients are shown in Table 1. With 50 simultaneous clients the FCTs increase being between 61 and 64 secs when using the small 2500 bytes router buffer. Not only the queueing delay increases compared to one and ten clients, but also little congestion starts to occur resulting in a few packet losses. The major reason for the increased FCT compared to one and 10 clients, however, is in increased queueing delay because on average only a few packets per client becomes dropped.

With 50 clients and a larger 14100 bytes router buffer, the buffer can absorb more packets eliminating all packet losses. This slightly increases the queueing delay because the packets dropped with the 2500 bytes buffer now fit into the buffer. This becomes visible as a slight increase in FCTs at lower percentiles. At the same time the FCT decreases for those flows for which no packet losses need to be recovered. This is visible as decreased FCTs at the upper percentiles and more stable FCTs compared to the case with 2500 bytes buffer.

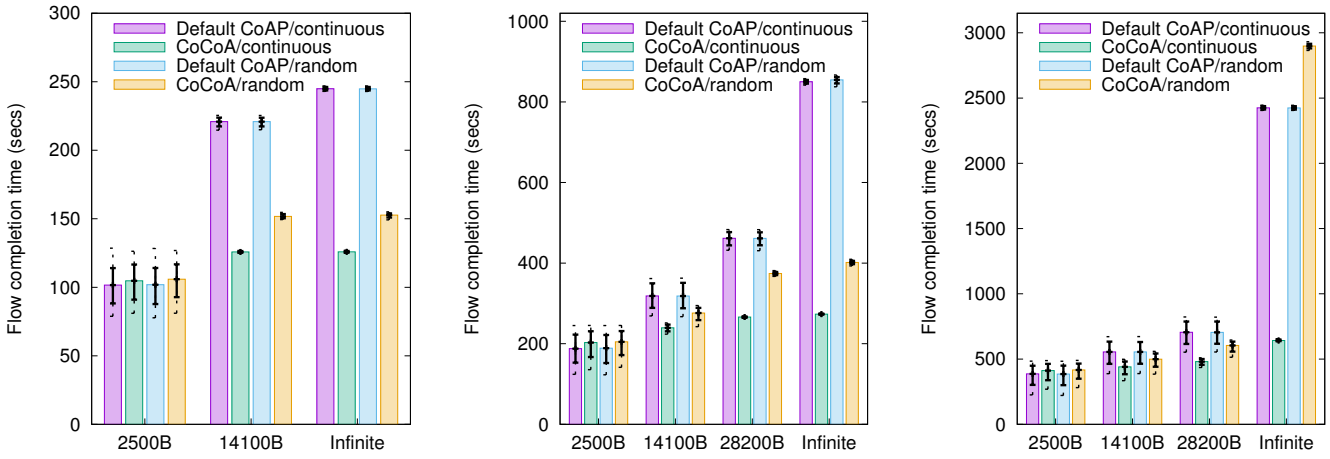
The differences between continuous and random clients are negligible with 50 clients as hardly any packet losses and retransmissions are present.

Figure 2a shows the median, quartiles, and 10th/90th percentiles of the FCT for 100 simultaneous clients of continuous and random type. The median FCTs increase substantially compared to the 50 simultaneous clients case. This is due to increased congestion that results in more packet losses with the small router buffer and longer queueing delay with the larger router buffers. The results with continuous and random clients are still very similar. With the 2500 bytes router buffer the median FCTs are approx. 102 seconds for Default CoAP and approx. 105 and 106 seconds for CoCoA with continuous and random clients, respectively.

With 100 clients and the smallest buffer both congestion control variants react to congestion by backing off the RTO when retransmitting a message. This allows exchanging most of the messages without retransmissions, while those clients for which retransmissions are needed suffer from long FCTs visible in the upper FCT percentiles. This indicates a Lock-Out phenomenon of some degree which is typical for a congested tail-drop router queue [1].

With 100 clients and larger buffer sizes, the buffers can absorb more packets eliminating most of the packet losses with the 14100 bytes buffer and all losses with the infinite buffer. Again, the larger buffer increases the queueing delay and results in longer FCTs for both congestion control variants. However, the queueing delay increases such that the round-trip time becomes extended well beyond 2 seconds and Default CoAP starts to show unacceptable behaviour because it is not able to adjust its RTO. Instead, it employs the initial RTO between 2 and 3 seconds for all message exchanges. When these messages hit the full queue many of them inevitably encounter a spurious retransmission timeout. The spurious RTOs further add to the load in the queue extending the RTT beyond 3 seconds and thereby result in one unnecessary retransmission for almost all request messages. Therefore, FCTs for Default CoAP are nearly twice as long as for CoCoA.

With 100 continuous clients and larger buffers CoCoA is able to adjust its RTO beyond the actual RTT after some number of message exchanges. However, during this time period it unnecessarily retransmits a few messages per client before it is able to update its RTO to a high enough value. The weak RTT samples often help CoCoA to update its RTO despite



(a) 100 clients (b) 200 clients (c) 400 clients

Figure 2: Flow completion times with different number of clients

of the retransmissions, but these unnecessary retransmissions further increase the queue size temporarily, slowing down the RTO adjustment. With random clients, however, the FCTs for CoCoA are higher compared to the FCTs with continuous clients, because the RTO is reset after each batch and CoCoA has problems to quickly inflate the RTO large enough to avoid unnecessary retransmissions. It, however, manages to acquire a few weak RTT samples that increase the RTO well above the initial RTO and therefore CoCoA unnecessarily retransmits clearly less than Default CoAP, resulting in a lower FCT than that of Default CoAP.

Figures 2b and 2c show the FCTs for 200 and 400 simultaneous clients. The higher congestion level results in further increase in the number of packet losses and/or queueing delay depending on the router buffer size. Also, FCTs are further extended as expected.

With the 2500 bytes buffer the median FCT for Default CoAP is slightly shorter than that of CoCoA. This is because CoCoA has some problems of adjusting RTO to a proper value as it takes weak RTT samples for retransmitted messages, resulting in somewhat higher RTO values compared to the initial RTO of 2-3 seconds that Default CoAP uses. Hence, on the average RTO expires a bit later for CoCoA than for Default CoAP, resulting in longer time to complete a message exchange when retransmissions are needed.

While Default CoAP copes reasonably well with the large number of clients and small router buffer that result in significant number of packet losses, it continues to indicate unsustainable behaviour with large buffer sizes. The queueing delay increases alongside the increase in buffer sizes. The larger queueing delay inherently affects the completion time of all clients, but leads to a vast number of spurious RTOs with Default CoAP due to the unadjustable RTO value in use. Even though Default CoAP applies exponential backoff to the timer on retransmissions, it does not help much at all as it is returned back to the initial RTO of two to three seconds for each new message exchange. This results in a significant amount of spurious RTOs. In particular, with 400 clients the large buffers are filled with unnecessary retransmissions and the capacity of the bottleneck link is wasted causing even more delay. This behaviour fulfils the symptoms of the classical congestion collapse [11], where little forward progress is made compared to the time spent in delivering the unnecessary retransmissions as indicated in Table 2. Default CoAP unnecessarily retransmits almost all messages four times, wasting nearly 80% of the bottleneck link capacity with unnecessary retransmissions.

CoCoA handles the increased congestion and delay with 200 continuous clients and larger buffers relatively well but with random clients FCTs for CoCoA increase further as the buffer size is increased, resulting in much higher FCTs for CoCoA with random clients than with continuous clients. This is due to increasing number of unnecessary retransmissions as CoCoA is not able to adjust its RTO promptly. With

Table 2: Frequency of retransmissions with 400 clients and infinite buffer (0=orig. transmission)

CC / Workload	0	1	2	3	4	5	6	7
Default/continuous	1474	3984	4793	7795	381954	0	0	0
CoCoA/continuous	339971	23563	19621	13216	3629	0	0	0
Default/random	1482	3995	4797	7965	381761	0	0	0
CoCoA/random	1982	14500	13915	8713	18510	245248	97132	0

400 random clients and the infinite buffer, however, the FCT for CoCoA explodes and becomes even higher than that of Default CoAP. The median FCT for CoCoA is approx. 2898 seconds and for Default CoAP 2425 seconds. With random clients CoCoA cannot timely adjust its RTO to the high RTT level caused by buffer-bloated queueing and it is forced to use the initial RTO values similar to Default CoAP, resulting in a huge number of unnecessary retransmissions. The number of unnecessary retransmissions for CoCoA is even higher than that of Default CoAP because CoCoA applies the variable backoff factor of 1.5 for most of the time, whereas Default CoAP employs the backoff factor of 2. This means that CoCoA random clients retransmit more aggressively and cause even higher degree of congestion collapse than Default CoAP clients. With random clients CoCoA unnecessarily retransmits almost all messages 5 to 6 times before getting a response as illustrated in Table 2.

A typical behaviour for a CoCoA client among 400 simultaneous random clients and with the infinite buffer size can be characterized as follows. The first random client may be able to get a weak RTT measurement for some of its message exchanges. A few individual weak RTT measurements, however, do not deviate the RTO much from the initial RTO value of 2 to 3 seconds, while the prevailing RTT is much higher and keeps increasing due to all unnecessary retransmissions. From this point on, each subsequent random client is unable get a valid RTT measurement as it is forced to retransmit each message more than two times. After several retransmissions it receives an acknowledgement but uses the initial RTO value of 2 to 3 seconds for the subsequent message and is again forced to retransmit several times. The same behaviour is repeated with all remaining random clients and all these unnecessary retransmissions explode the queueing delay and raise the perceived round-trip time drastically.

The results indicate that neither Default CoAP nor CoCoA are able to scale properly with increasing level of congestion, resulting in congestion collapse in the worts case. They both fail to respond properly to congestion as they revert the backed off retransmission timer back to a too low RTO value immediately after getting an acknowledgement for an unnecessarily retransmitted message. In addition, both algorithms have problems in adjusting the RTO. Default CoAP is not able to adjust the RTO at all but always uses an initial RTO of 2-3 seconds for a new message exchange, while CoCoA uses a blind initial RTO of 2 seconds to initialize the RTO, instead of using a proper RTT sample when available. CoCoA also uses RTO calculation that adjusts the RTO very slowly because it applies an additional weight of 0.5 or 0.25 when computing the RTO.

4.2 Full Backoff Congestion Control

We implement a couple of simple modifications for both Default CoAP and CoCoA congestion control to make them congestion safe. We call these Full Backoff variants.

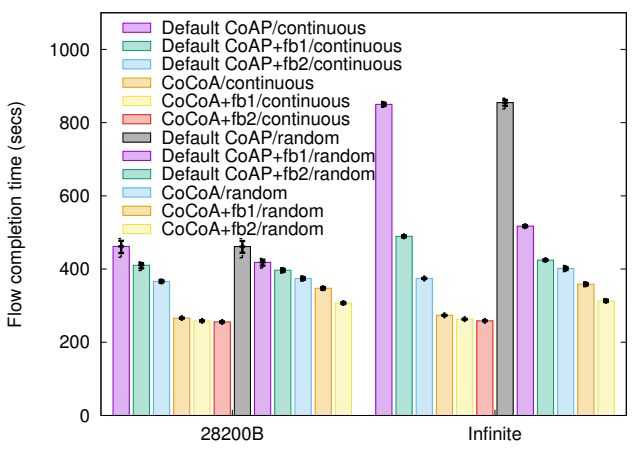
For Default CoAP we first implement the *Fullbackoff1* variant that is similar to Karn's algorithm; it retains, after retransmitting, the backed off RTO value for the subsequent message exchange until an exchange with no retransmissions is successfully completed. If the already backed off timer expires with the subsequent exchange, the RTO value is doubled as usual. After a successful exchange with no retransmissions, the initial RTO is reverted.

The *Fullbackoff1* variant, however, leaves Default CoAP quite aggressive in high latency environments because it always starts over with the preset initial RTO after a successful, non-retransmitted message exchange. Therefore, we introduce a more conservative *Fullbackoff2* variant for Default CoAP that, instead of reverting the initial RTO after a successful exchange, halves the backed off RTO after each successful exchange until the initial RTO value is reached. This results in more conservative and thereby more congestion safe behaviour when no RTT measurement and RTO adjustment is present.

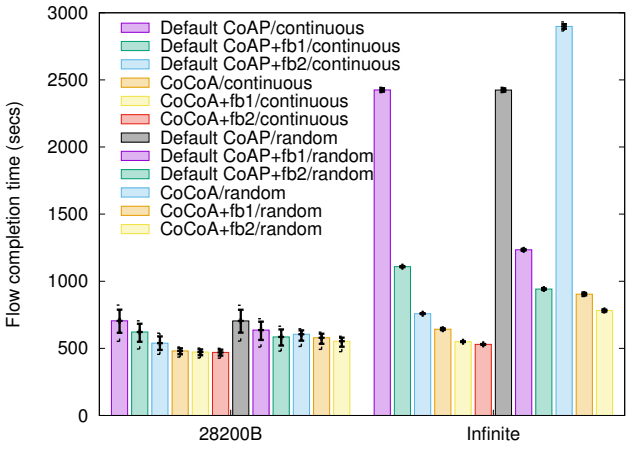
CoCoA *Fullbackoff1* variant retains the backed off RTO value after retransmissions like the *Fullbackoff1* variant for Default CoAP. This approach, however, does not take into account that CoCoA may update the weak RTO estimate after retransmitting. This update, may increase the RTO value significantly, even beyond the currently backed off timer value, in case the actual RTT is rapidly increasing. To address this, we implement also the *Fullbackoff2* variant for CoCoA that, after retransmitting, takes the maximum of the current RTO and a newly updated RTO, if available. It then recalculates the backed off RTO based on the maximum and uses it for the next message exchange.

The Flow completion times (FCTs) for 200 and 400 clients with the Full Backoff variants are shown in Figures 3a and 3b. The Fullbackoff variants for both Default CoAP and CoCoA reduce the FCTs with larger buffer sizes, yielding a significant decrease in FCTs with the infinite buffer, particularly when using 400 random clients.

The effectiveness of the Full Backoff variants is further depicted in Figure 4 that shows a dramatic decrease in the number of unnecessary retransmissions with the bufferbloated bottlenecks. When running continuous clients, the Default CoAP *Fullbackoff1* variant yields clearly more conservative behaviour than the existing Default CoAP, reducing the median for the number of unnecessary retransmissions per client by 68% and 47% with the infinite and 28200 bytes buffer, respectively. With random clients, the reduction in the number of unnecessary retransmissions per client is slightly less



(a) 200 clients



(b) 400 clients

Figure 3: Flow completion times with full backoff

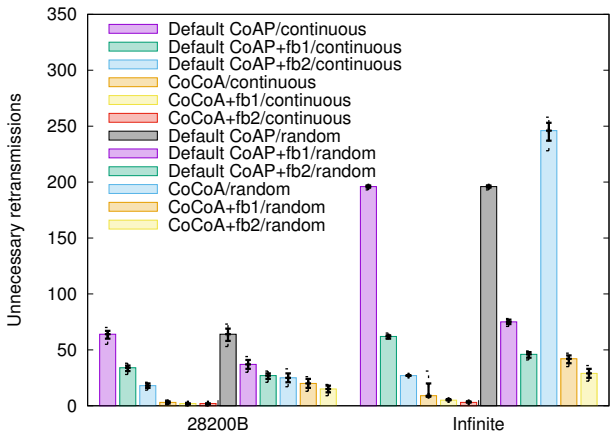


Figure 4: The number of unnecessary retransmissions per client with 400 clients and full backoff

significant because a new random client using the initial RTO starts relatively often and retransmissions with the initial RTO cannot be avoided. Nevertheless, the median for the number of unnecessary retransmissions is reduced by 62% with the infinite buffer and by 42% with the 28200 bytes buffer.

The *Fullbackoff2* variant for Default CoAP yields even more notable reduction in the number of unnecessary retransmissions per client compared to the existing Default CoAP. For *Fullbackoff2* variant the median of the unnecessary retransmissions per client with the infinite buffer and 28200 bytes buffer is 86% and 72% less than that of the existing Default CoAP, respectively. With random clients, the *Fullbackoff2* variant reduces the median for the number of unnecessary retransmissions compared to the existing Default CoAP by 77% and 58% with the infinite buffer and 28200 bytes buffer, respectively.

The Full Backoff variants for CoCoA avoid the congestion collapse with random clients and infinite buffer effectively. The median for the number of unnecessary retransmissions per client is reduced by 83% and by 88% with the *Fullbackoff1* and *Fullbackoff2* variant, respectively.

We also run a set of experiments where different CoAP endpoints implement different congestion control variants to confirm that the misbehaviour of the existing congestion controls also have an adverse effect on competing traffic. As expected, the results confirm that when a half of the clients runs Default CoAP and the other half runs one of the Full Backoff variants misbehaving Default CoAP ruins the performance of the well-behaving variant, while at the same time the well-behaving variant helps misbehaving Default CoAP to improve its performance.

5 CONCLUSIONS

In this paper we evaluate the Default CoAP and CoCoA congestion control algorithms in environments with varying levels of congestion and bufferbloat. The results indicate that both algorithms perform a large number of unnecessary retransmissions in a bufferbloated environment and result in wasting most of the network capacity similar to classical congestion collapse. We identify the root cause for the issue in the robustness of the backoff logic and propose improved backoff algorithms to mitigate the issue. The evaluation of the improved algorithms demonstrate their effectiveness.

REFERENCES

[1] F. Baker and G. Fairhurst. 2015. *IETF Recommendations Regarding Active Queue Management.* RFC 7567.

[2] A. Betzler, C. Gomez, I. Demirkol, and M. Kovatsch. 2014. Congestion Control for CoAP Cloud Services. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA).* 1–6. https://doi.org/10.1109/ETFA.2014.7005340

[3] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells. 2015. CoCoA+: An Advanced Congestion Control Mechanism for CoAP. *Ad Hoc Networks* 33 (2015), 126–139. https://doi.org/10.1016/j.adhoc.2015.04.007

[4] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells. 2016. CoAP Congestion Control for the Internet of Things. *IEEE Communications Magazine* 54, 7 (July 2016), 154–160. https://doi.org/10.1109/MCOM.2016.7509394

[5] C. Bormann, A. Betzler, C. Gomez, and I. Demirkol. 2018. *CoAP Simple Congestion Control/Advanced.* Internet Draft. https://www.ietf.org/id/draft-ietf-core-cocoa-03.txt Work in progress.

[6] S. Farrell. 2018. *Low-Power Wide Area Network (LPWAN) Overview.* RFC 8376.

[7] J. Gettys and K. Nichols. 2011. Bufferbloat: Dark Buffers in the Internet. *ACM Queue* 9, 11 (Nov. 2011). https://doi.org/10.1109/MIC.2011.56

[8] I. Järvinen, L. Daniel, and M. Kojo. 2015. Experimental Evaluation of Alternative Congestion Control Algorithms for Constrained Application Protocol (CoAP). In *The 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT).* https://doi.org/10.1109/WF-IoT.2015.7389097

[9] P. Karn and C. Partridge. 1987. Improving Round-trip Time Estimates in Reliable Transport Protocols. In *SIGCOMM'87 Proceedings of the ACM Workshop on Frontiers in Computer Communications Technology.* 2–7. https://doi.org/10.1145/55482.55484

[10] Libcoap [n. d.]. libcoap: C-Implementation of CoAP. Retrieved June 15, 2018 from https://libcoap.net/

[11] J. Nagle. 1984. *Congestion Control in IP/TCP Internetworks.* RFC 896.

[12] V. Paxson, M. Allman, J. Chu, and M. Sargent. 2011. *Computing TCP's Retransmission Timer.* RFC 6298.

[13] U. Raza, P. Kulkarni, and M. Sooriyabandara. 2017. Low Power Wide Area Networks: An Overview. *IEEE Communications Surveys Tutorials* 19, 2 (Jan. 2017), 855–873. https://doi.org/10.1109/COMST.2017.2652320

[14] Z. Shelby, K. Hartke, and C. Bormann. 2014. *The Constrained Application Protocol (CoAP).* RFC 7252.

[15] F. Zheng, B. Fu, and Z. Cao. 2016. *CoAP Latency Evaluation.* Internet Draft. https://www.ietf.org/archive/id/draft-zheng-core-coap-lantency-evaluation-00.txt Work in progress.